Inheritance Solutions

Copyright © James Raynard 2021

Inheriting constructors

- What is meant by an "inheriting constructor"?
  - In a derived class, we can put a "using" statement for the base class's constructor
  - This will cause the compiler to add a constructor which has the same arguments as the base class's constructor
  - This inheriting constructor will call the base class's constructor, forwarding its arguments
  - If the base class has multiple constructors, the compiler will inherit all of them

Copyright © James Raynard 2021

Inheriting constructors

- Why are inheriting constructors useful?
  - They avoid the need to write tedious, error-prone "boilerplate" code
- If the derived class has a member which is not in the base class, how can this be initialized?
  - Use an in-place initializer
- Write a simple program which demonstrates the use of inheriting constructors

Copyright © James Raynard 2021

override keyword

- What is the override keyword?
  - We can declare a member function in a derived class as "override"
  - The compiler will check whether it overrides a virtual function in the base class
  - If not, there is a compiler error
- Why is it useful?
  - The compiler will detect cases in which the derived class function has the wrong arguments, or the base class function is not virtual
  - This helps catch editing mistakes
- Write a simple program to demonstrate the use of the override keyword

Copyright © James Raynard 2021

final keyword

- What effect does declaring a member function "final" have?
  - The member function cannot be overridden in a derived class
- What effect does declaring a class "final" have?
  - The class cannot be used as a base class, i.e. it cannot be derived from
- Why is the final keyword useful?
  - It can be used to write a library which uses inheritance internally, but does not allow users of the library to extend the inheritance hierarchy
- Write a simple program which demonstrates the use of the final keyword

Copyright © James Raynard 2021